CLAIMS

1. Method for realising a management activity of at least one managed object (B1, ..., BN) by at least one manager object through a communication network (R), characterised in that it comprises the following steps:

- providing at least one intermediate object (AG) configured to manage said at least one managed object (B1, ..., BN) according to a data set (1102), said management activity being transformed into a set of results (1112),
- providing said data set (1100) from said at least one manager object (A) to said intermediate object (AG),
- managing said at least one managed object (B1, ..., Bn) through said at least one intermediate object (AG), to generate said set of results,
- transferring (1108) said set of results from said at least one intermediate object (AG) to said at least one manager object (A).

2. Method according to claim 1, characterised in that it comprises the step of establishing the communication between said at least one manager object (A) and said at least one intermediate object via UDP protocol.

3. Method according to claim 1 or claim 2, characterised in that it comprises the following steps:

- managing at least one further managed object (Bk, ..., BN) directly through said at least one manager object (A), and
- managing said at least one managed object (B1, B2, B3) by said at least one manager object (A) via said intermediate object (AG).

4. Method according to claim 3, characterised in that it comprises the management of said at least one further managed object (Bk, ..., Bn) and said at least one managed object (B1, B2, B3) through a single communication network (R).

5. Method according to claim 3, characterised in that it comprises the following steps:

- providing a first communication network (RP) for managing said at least one further managed object (B1) directly through said at least one manager object (A) and transferring said data set (1100) and said results set (1118) between said at least one manager object (A) and said at least one further managed object (B1), and

- providing a second communication network (RA) for managing said at least one managed object (B2, B3) through said intermediate object (AG).

6. Method according to any of the previous claims, characterised in that it comprises the steps of providing a plurality of said intermediate objects (AG1, AG2) and managing at least one managed object (B3) through several intermediate objects (AG1, AG2) of said plurality.

7. Method according to any of the previous claims, characterised in that said intermediate object (AG) is provided with respective reception modules (ARX) and transmission modules (ATX) configured so that said at least one manager object (A) sees said intermediate object (AG) essentially as one of said managed objects (B1, ..., Bn).

8. Method according to any of the previous claims, characterised in that said at least one intermediate object (AG) comprises at least one respective management module (MM) configured so that said at least one managed object (B1, ..., BN), which is managed by said at least one intermediate object (AG), sees said at least one intermediate object (AG) essentially as said at least one manager object (A).

9. Method according to any of the previous claims, characterised in that said at least one intermediate object (AG) is provided with one of the following queues:

- an input queue (I) for collecting input messages with

respect to said at least one intermediate object (AG),

- an output queue (U) for collecting output messages from said at least one intermediate object (AG), and

- a working queue (L) for collecting messages inherent to said management activity performed by said at least one intermediate object (AG) on said at least one managed object (B1, ..., Bn).

10. Method according to claim 9, characterised in that it comprises the step of providing, in said at least one intermediate object (AG), a dedicated module (DC) for analysing the input messages received by said input queue (I).

11. Method according to claim 9 or claim 10, characterised in that it comprises the following steps:

- providing, in said at least one intermediate object (AG), an activity co-ordinating module (CA) for implementing at least one of the following functions:

- instanctiating at least one concurrent process,

- updating activity status of the requests in said working queue L, and

- creating statistic check messages to be sent to said at least one manager object (A) through said output queue (U).

12. Method according to any of the previous claims, characterised in that it comprises the step of providing a plurality of protocol management modules (MP1, MP2, MP3) configured to establish communication to said at least one managed object (B1, ..., BN) through respective different protocols in said at least one intermediate object (AG).

13. Method according to any of the previous claims, characterised in that it comprises the step of establishing the communication between said at least one manager object (A) and said at least one intermediate object (AG) by subjecting at least one part of the respective messages to

a compression operation (302; 104, 204).

14. Method according to claim 13, characterised in that said compression operation is based on the acknowledgement of a sequence which appears periodically in the message.

15. Method according to claim 14, characterised in that said compression operation implements a gzip type method, such as zLib.

16. Method according to claim 2 and any of the claims from 13 to 15, characterised in that it comprises the step of indicating that compression of the message transferred by UDP is done.

17. Method according to claim 16, characterised in that a bit field in the UDP header is used to indicate that the compression operation (302) is done.

18. Method according to claim 17, characterised in that bits comprised in the range from bit 62 to bit 69 in the UDP header are used in indicate that the compression operation (302) is done.

19. Method according to claim 18, characterised in that comprises the step of setting at least one of the bits from 62 to 69 of the UDP message header to 1.

20. Method according to any of the claims from 13 to 19, characterised in that the communication between said at least one manager object (A) and said at least one intermediate object (AG) is implemented by means of SNMP messages, and comprises the following steps during the compression step:

- reading (100) the entire SNMP message,
- encoding (102) the read message in hexadecimal format, and
- subjecting the message encoded in hexadecimal format to compression (104).

21. Method according to any of the claims from 13 to 19, characterised in that communication between said at

least one manager object (A) and said at least one intermediate object (AG) is implemented by means of SNMP messages, comprises the following steps during the reception step:

- subjecting the received message to decompression (204) complementary to said compression operation, to obtain a message subjected to decoding in hexadecimal format,
- decoding (202) the message from the hexadecimal format, and
- reconstructing (200) the entire SNMP message from said decoded message.

22. Method according to claim 20 or claim 21, characterised in that it comprises a nesting operation in a standard SNMP message for the transmission of the message subjected to said compression operation (104).

23. Method according to claim 22, characterised in that it comprises the following steps during transmission:

- reading (108) the message subjected to said compression operation (104) in bytes and transposing (110) it into a corresponding ASCII character message,
- generating (112) a variable binding set comprising a first OID indicating the original file size and subsequent OID/value pairs which carry portions of said message subjected to said compression operation (104) transposed into ASCII characters,
- reconstructing SNMP message header data,
- encoding (114) the resulting SNMP message in hexadecimal format to generate the UDP payload, and
- transferring (116) the UDP payload generated in this way.

24. Method according to claim 22 or claim 23, characterised in that it comprises the following steps during reception:

- receiving the message subjected to said compression operation as an UDP payload (216),

- subjecting the payload received in this way to a hexadecimal decoding operation (214),

- acknowledging and assembling (212) the variable binding of the message subjected to hexadecimal decoding,

- subjecting the message subjected to said acknowledging and assembling operation (212) to binary ASCII decoding (210), and

- subjecting the decoded message in binary form to said decompression operation (204).

25. Method according to claim 20 or claim 21, characterised in that it comprises the step of integrating the message subjected to said compression operation (104) through UDP nesting for the transmission of the message subjected to said compression operation (104).

26. Method according to claim 25, characterised in that it comprises the following steps during transmission:

- configuring said message subjected to said compression operation (104) as a Protocol Data Unit (PDU) payload, and

- transferring the payload created in this way to a given receiver port.

27. Method according to claim 25 or claim 26, characterised in that it comprises the following steps during reception:

- receiving said message as a payload of a PDU UDP received at a receiver port, and

- extracting said payload from said PDU.

28. Method according to claim 26 or claim 27, characterised in that it comprises the step of transmitting a synchronisation message (1106) of the SNMP type indicating said transmission port and/or said reception port between said at least one manager object (A) and said

at least one intermediate object (AG).

29. System for managing communication networks comprising at least one manager object (A) and at least one managed object (B1, ..., Bn), characterised in that it comprises at least one intermediate object (AG) implementing the method according to any of the claims from 1 to 28.

30. Software modules which can be directly loaded into the internal memory of at least a computer and comprising portions of software code to implement the method according to any of the claims from 1 to 28 when the software modules are run by at least one computer.